UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/661,982	09/12/2003	Cary Lee Bates	ROC920000051.D1	9327

46797 7590 03/27/2008

IBM CORPORATION, INTELLECTUAL PROPERTY LAW
DEPT 917, BLDG. 006-1
3605 HIGHWAY 52 NORTH
ROCHESTER, MN 55901-7829

EXAMINER	
WANG, BEN C	

ART UNIT	PAPER NUMBER
2192	

MAIL DATE	DELIVERY MODE
03/27/2008	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	10/661,982	BATES ET AL.	
	Examiner	Art Unit	
	BEN C. WANG	2192	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE *3* MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *07 January 2008*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-22* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-22* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. Applicant's amendment dated January 7, 2008, responding to the Office action mailed October 5, 2007 provided in the rejection of claims 1-22, wherein claims 14-21 have been amended.

Claims 1-22 remain pending in the application and which have been fully considered by the examiner.

Applicant's arguments with respect to claims currently amended have been fully considered but are moot in view of the new grounds of rejection – see *Sparks et al.* - art made of record, as applied hereto.

Claim Rejections – 35 USC § 102(b)

The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102(b) that form the basis for the rejections under this section made in this office action:

> A person shall be entitled to a patent unless –
>
> (b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

2. Claims 1-8, 11, 14-16, 19, and 22 are rejected under 35 U.S.C. 102(b) as being anticipated by Sparks et al. (Pat. No. 5,355,469) (hereinafter 'Sparks' - art made of record)

3. **As to claim 1** (Original), Sparks discloses a method for managing memory available for dynamic allocation during execution of code containing a plurality of memory allocators and a plurality of memory deallocators, comprising:

- allowing a user to establish a relationship between one or more of the memory deallocators and one or more of the memory allocators, wherein the relationship requires that memory space allocated by the one or more allocators is freed by the one or more deallocators (e.g., Col. 3, Lines 14-31 - … uses a pre-defined first table structure to monitor all data storage allocations and deallocations of a computer program. For each allocation request, a check is made as to whether the request passes certain validity checks and does not conflict with an existing allocation in the table structure. If no verification problem exists, an entry is made in the first table for the pending allocation request … if a pending memory allocation violates a validity check or conflicts with an existing allocation, an error message is generated … For each deallocation request, a check is made as to whether the request passes certain validity checks and a search of the first table is made to verify if a previous allocation has been made. If the request is valid … Otherwise, an error message is generated);
- allowing the code to execute; upon a call to the one or more deallocators to free a memory space, determining whether the relationship is violated (e.g., Abstract - … automatically detecting errors in computer program caused by erroneous memory allocations and deallocations … uses a predefined first table structure to

monitor all data storage allocations and deallocations of a computer program …); and

- if so, notifying the user (e.g., Col. 3, Lines 52-54 - … error messages may be directed to .... for automatic handling memory allocation/deallocation errors).

4. **As to claim 2** (Original) (incorporating the rejection in claim 1), Sparks discloses the method wherein notifying the user comprises halting execution of the code (e.g., Col. 8, Lines 55-56 - … generates an error message, execution of the program can be halted).

5. **As to claim 3** (Original) (incorporating the rejection in claim 1), Sparks discloses the method wherein notifying the user comprises halting execution of the code and displaying a status message to the user (e.g., Col. 8, Lines 55-56 - … generates an error message, execution of the program can be halted; Col. 8, Lines 30-34 - … outputs to the user (for example, … screen display …) the variable name locations of the variables that were not deallocated).

6. **As to claim 4** (Original) (incorporating the rejection in claim 1), Sparks discloses the method if the relationship is not violated, freeing the memory space (e.g., Col. 3, Lines 14-31 - … uses a pre-defined first table structure to monitor all data storage allocations and deallocations of a computer program … For each deallocation request, a check is made as to whether the request passes certain validity checks and a search of the first table is made to verify if a previous allocation has been made. If the request

is valid, the corresponding entry in the first table is deleted, and processing continues in a normal fashion …)

7. **As to claim 5** (Original) (Original) (incorporating the rejection in claim 1), Sparks discloses the method wherein determining whether the relationship is violated comprises determining that the memory space was allocated by an allocator different from the one or more memory allocators (e.g., Abstract - … automatically detecting errors in computer program caused by erroneous memory allocations and deallocations … uses a predefined first table structure to monitor all data storage allocations and deallocations of a computer program …)

8. **As to claim 6** (Original), Sparks discloses a method for managing memory available for dynamic allocation during execution of code containing a plurality of memory allocators and a plurality of memory deallocators, comprising:

- establishing a relationship between a user-selected memory deallocator and a user-selected memory allocator, wherein the relationship requires that memory space freed by the user-selected deallocator have been allocated by the user-selected allocator (e.g., Col. 3, Lines 14-31 - … uses a pre-defined first table structure to monitor all data storage allocations and deallocations of a computer program. For each allocation request, a check is made as to whether the request passes certain validity checks and does not conflict with an existing allocation in the table structure. If no verification problem exists, an entry is made in the first

table for the pending allocation request … if a pending memory allocation violates a validity check or conflicts with an existing allocation, an error message is generated … For each deallocation request, a check is made as to whether the request passes certain validity checks and a search of the first table is made to verify if a previous allocation has been made. If the request is valid … Otherwise, an error message is generated);

- allowing the code to execute;
- upon a call to the user-selected deallocator to free a memory space, determining whether the memory space was allocated by the user-selected allocator (e.g., Abstract - … automatically detecting errors in computer program caused by erroneous memory allocations and deallocations … uses a predefined first table structure to monitor all data storage allocations and deallocations of a computer program …); and
- if so, notifying the user that the relationship is violated (e.g., Col. 3, Lines 52-54 - … error messages may be directed to .... for automatic handling memory allocation/deallocation errors))

9. **As to claim 7** (Original) (incorporating the rejection in claim 6), please refer to claim **3** as set forth above accordingly.

10. **As to claim 8** (Original), Sparks discloses a method for managing memory available for dynamic allocation during execution of code containing a plurality of memory allocators and a plurality of memory deallocators (e.g.,), comprising:

- setting an upper limit on the amount of memory space an allocator can allocate during execution of the code, wherein the upper limit is specific to the allocator (e.g., Col. 5, Lines 62-65 - … set by the user to be the maximum number of concurrent memory allocation …);
- during execution of the code, tracking the amount of memory space allocated by the allocator (e.g., Abstract - … automatically detecting errors in computer program caused by erroneous memory allocations and deallocations … uses a predefined first table structure to monitor all data storage allocations and deallocations of a computer program …); and
- when the amount of memory space allocated exceeds the limit, notifying a user (e.g., Col. 3, Lines 52-54 - … error messages may be directed to .... for automatic handling memory allocation/deallocation errors).

11. **As to claim 11** (Original) (incorporating the rejection in claim 8), Sparks discloses notifying the user comprises halting execution of the code (e.g., Col. 8, Lines 55-56 - … generates an error message, execution of the program can be halted)

12. **As to claim 14** (Currently Amended), Sparks discloses a computer readable storage medium containing a program which, when executed, performs an operation for managing memory available for dynamic allocation during execution of code containing a plurality of memory allocators and a plurality of memory deallocators, the operation comprising:

- establishing a relationship between a user-selected memory deallocator and a user-selected memory allocator, wherein the relationship requires that memory space freed by the user-selected deallocator have been allocated by the user-selected allocator (e.g., Col. 3, Lines 14-31 - … uses a pre-defined first table structure to monitor all data storage allocations and deallocations of a computer program. For each allocation request, a check is made as to whether the request passes certain validity checks and does not conflict with an existing allocation in the table structure. If no verification problem exists, an entry is made in the first table for the pending allocation request … if a pending memory allocation violates a validity check or conflicts with an existing allocation, an error message is generated … For each deallocation request, a check is made as to whether the request passes certain validity checks and a search of the first table is made to verify if a previous allocation has been made. If the request is valid … Otherwise, an error message is generated);
- allowing the code to execute;
- upon a call to the user-selected deallocator to free a memory space, determining whether the memory space was allocated by the user-selected allocator (e.g., Abstract - … automatically detecting errors in computer program caused by erroneous memory allocations and deallocations … uses a predefined first table structure to monitor all data storage allocations and deallocations of a computer program …); and

- if so, notifying the user that the relationship is violated (e.g., Col. 3, Lines 52-54 - … error messages may be directed to .... for automatic handling memory allocation/deallocation errors).

13. **As to claim 15** (Currently Amended) (incorporating the rejection in claim 1), please refer to claim **3** as set forth above accordingly.

14. **As to claim 16** (Currently Amended), Sparks discloses a computer readable storage medium containing a program which, when executed, performs an operation for managing memory available for dynamic allocation during execution of code containing a plurality of memory allocators and a plurality of memory deallocators, the operation comprising:

- setting an upper limit on the amount of memory space an allocator can allocate during execution of the code, wherein the upper limit is specific to the allocator (e.g., Col. 5, Lines 62-65 - … set by the user to be the maximum number of concurrent memory allocation …);
- during execution of the code, tracking the amount of memory space allocated by the allocator (e.g., Abstract - … automatically detecting errors in computer program caused by erroneous memory allocations and deallocations … uses a predefined first table structure to monitor all data storage allocations and deallocations of a computer program …); and

- when the amount of memory space allocated exceeds the limit, notifying a user (e.g., Col. 3, Lines 52-54 - … error messages may be directed to .... for automatic handling memory allocation/deallocation errors).

15. **As to claim 19**, please refer to claim **11** as set forth above accordingly.

16. **As to claim 22** (Original), Sparks discloses a computer system comprising an output device, a memory device, one or more processors, code resident in the memory device and containing a plurality of memory allocator calls and a plurality of memory deallocator calls, a heap manager resident in the memory device to allocate and free memory of the memory device and a debugger program resident in the memory device; the debugger program comprising a debugger user interface configured to at least:

- allow a user to view allocation/deallocation history information at a user-specified memory location (e.g., Col. 3, Lines 32-35 - … to track existing memory allocations that are not properly deallocated …; Col. 5, Lines 57-60 - … for tracking memory blocks that have not been properly allocated …; Col. 5, Line 65 through Col. 6, Line 1 - … to track that memory block is made available for reuse for a subsequent memory allocation); and
- allow a user to establish a relationship between a memory deallocator call and a memory allocator call, wherein the relationship requires that memory space allocated by the memory allocator call is freed by the memory deallocator call and a violation of the requirement causes the debugger user interface to notify

the user (e.g., Abstract - … automatically detecting errors in computer program caused by erroneous memory allocations and deallocations … uses a predefined first table structure to monitor all data storage allocations and deallocations of a computer program … Col. 3, Lines 52-54 - … error messages may be directed to .... for automatic handling memory allocation/deallocation errors; Col. 8, Lines 50-54 – this permits a program developer to step back through the program code with a standard debugging program … after the code line that attempted to allocate memory …).

Claim Rejections – 35 USC § 103(a)

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

17. Claims 9-10, 12-13, 17-18, and 20-21 are rejected under 35 U.S.C. 103(a) as being unpatentable over Sparks in view of Sato et al. (Pat. No. US 6,467,075 B1) (hereinafter 'Sato')

18. **As to claim 9** (Original) (incorporating the rejection in claim 8), Sparks does not explicitly disclose the step of tracking comprises:

- determining whether the allocator is called to allocate memory and, if so, incrementing a counter; and
- determining whether a deallocator is called to deallocate memory allocated by the allocator and, if so, decrementing the counter.

However, in an analogous art of *Resolution of Dynamic Memory Allocation/Deallocation and Pointers*, Sato discloses the step of tracking comprises:

- determining whether the allocator is called to allocate memory and, if so, incrementing a counter; and
- determining whether a deallocator is called to deallocate memory allocated by the allocator and, if so, decrementing the counter (e.g., Col. 17, Lines 59-67 – the idea is to have a counter for each dynamically-allocated location set; during the analysis, the counter is incremented each time an element of the corresponding location set is allocated; subsequently, each time an element of the location set is deallocated, the associated counter is decremented; this way, location sets allocated and not deallocated within these locations cannot be optimized; otherwise, they can be optimized)

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Sato into the Sparks' system to further provide the followings in the Sparks system:

- determining whether the allocator is called to allocate memory and, if so, incrementing a counter; and

- determining whether a deallocator is called to deallocate memory allocated by the allocator and, if so, decrementing the counter.

The motivation is that it would further enhance the Sparks' system by taking, advancing and/or incorporating the Sato's system which offers significant advantages for efficiently mapping arbitrary C code with pointers and malloc/free into hardware as once suggested by Sato (e.g., Abstract, Lines 10-13)

19. **As to claim 10** (Original) (incorporating the rejection in claim 8), Sato discloses the step of tracking comprises incrementing a counter in the event of memory allocation by the allocator and decrementing the counter in the event of memory deallocation of memory space allocated by the allocator (e.g., Col. 17, Lines 59-67 – the idea is to have a counter for each dynamically-allocated location set; during the analysis, the counter is incremented each time an element of the corresponding location set is allocated; subsequently, each time an element of the location set is deallocated, the associated counter is decremented; this way, location sets allocated and not deallocated within these locations cannot be optimized; otherwise, they can be optimized)

20. **As to claim 12** (Original) (incorporating the rejection in claim 8, Sato discloses wherein the upper limit is independent of other memory size limitations (e.g., Col. 9, Line 41 through Col. 10, Line 29 – e.g., allocate memory in *local_RAM* or allocate memory in *shared_RAM*; Col. 14, Line 64 through Col. 15, Line 29 – since the size of the dynamically allocated memory is a priori unknown at compile time, the designer also

sets the size of each memory segment; the tool instantiates then the allocators corresponding to each memory segment …; for each memory segment, a different allocator is instantiated; each malloc mapped to this memory segment is then replaced by a call to the specific allocator; the inventors generate a branching statement in which the different allocators corresponding the different memory segments may be called according to the pointer's tag; the pointer's index is then sent to the allocator to indicate which block should be deallocated)

21. **As to claim 13** (Original) (incorporating the rejection in claim 8), Sato discloses wherein the upper limit is not a limit on a stack size (e.g., Col. 9, Line 41 through Col. 10, Line 29 – e.g., allocate memory in *local_RAM* or allocate memory in *shared_RAM*; Col. 14, Line 64 through Col. 15, Line 29 – since the size of the dynamically allocated memory is a priori unknown at compile time, the designer also sets the size of each memory segment; the tool instantiates then the allocators corresponding to each memory segment …; for each memory segment, a different allocator is instantiated; each malloc mapped to this memory segment is then replaced by a call to the specific allocator; the inventors generate a branching statement in which the different allocators corresponding the different memory segments may be called according to the pointer's tag; the pointer's index is then sent to the allocator to indicate which block should be deallocated)

22. **As to claims 17-18**, please refer to claims **9-10** as set forth above accordingly.

23. **As to claims 20-21**, please refer to claims **12-13** as set forth above accordingly.

Conclusion

24. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Ben C. Wang whose telephone number is 571-270-1240. The examiner can normally be reached on Monday - Friday, 8:00 a.m. - 5:00 p.m., EST.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Q. Dam can be reached on 571-272-3695. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/Ben C Wang/
Examiner, Art Unit 2192
March 21, 2008

/Tuan Q. Dam/
Supervisory Patent Examiner, Art Unit 2192